Jurnal SIMETRIS, Vol. 10 No. 2 November 2019
P-ISSN: 2252-4983, E-ISSN: 2549-3108

PENGEMBANGAN APLIKASI OTOMATISASI ADMINISTRASI JARINGAN BERBASIS WEBSITE MENGGUNAKAN BAHASA PEMROGRAMAN PYTHON

Rheza Adhyatmaka Wiryawan
Departemen Teknik Elektro dan Informatika, Sekolah Vokasi
Universitas Gadjah Mada
Email: rheza.adhyatmaka.w@mail.ugm.ac.id

Nur Rohman Rosyid
Departemen Teknik Elektro dan Informatika, Sekolah Vokasi
Universitas Gadjah Mada
Email: nrohmanr@ugm.ac.id

ABSTRAK

Banyaknya perangkat jaringan yang terpasang merupakan sebuah tantangan bagi perusahaan. Semua perangkat yang terpasang perlu dilakukan pemeliharaan dan konfigurasi jaringan secara berkala agar jaringan dapat berjalan dengan baik dan tidak menghambat proses bisnis perusahaan. Pada cara tradisional administrator jaringan perlu masuk ke sistem perangkat secara satu persatu sehingga akan memakan waktu yang lama dan kurang efisien. Otomatisasi jaringan merupakan solusi untuk melakukan pekerjaan-pekerjaan yang rumit dan repetitif tersebut. Pekerjaan yang bersifat repetitif seperti *backup* konfigurasi, *restore* konfigurasi, dan lain-lain dapat dilakukan otomatisasi. Proyek akhir ini membuat sistem aplikasi otomatisasi administrasi jaringan berbasis *web*. Pengembangan aplikasi menggunakan metode *Rapid Application Development* untuk mengidentifikasi persyaratan aplikasi, perancangan, pembuatan dan implementasi aplikasi. Sistem aplikasi memanfaatkan *library* utama Paramiko sebagai penghubung dan otomatisasi jaringan dari *server* ke perangkat jaringan menggunakan protokol SSHv2 dan *framework* Django sebagai pengembangan *web*. Pengujian yang dilakukan terhadap aplikasi menggunakan metode *Black-Box Testing*. Hasil dari proyek akhir ini aplikasi dapat dimanfaatkan sebagai otomatisasi jaringan dalam hal konfigurasi *routing static, dynamic*, pembuatan VLAN, *maintenance* berupa *backup* dan *restore* secara terpusat sehingga akan lebih termanajemen lebih baik.

Kata kunci: otomatisasi jaringan; python; django; paramiko; manajemen jaringan.

ABSTRACT

The number of network devices installed becomes a challenge for network administrator in a company. All installed network devices need to be regularly maintained and configured to make sure the network working properly and does not interrupt the company's business processes. In the traditional method, network administrator do these job manualy for each network device that take a long time and be less efficient. An automation of network configuration is an alternative solution for doing complex, repetitive, and rutin jobs. This final project creates a web-based application of network administration automation using Python programming. Software development using Rapid Application Development method for identification, design, develop, and implementation. A software library of Paramiko is utilised as a link and an automation engine from a server to network devices using SSHv2 protocol, and the Django framework roles as a web engine. A Black-Box testing method is used to validate all functionalities of the application. A contribution of the final project, gives an alternative solution for automation of network configuration, routing static, dynamic, create VLAN, backup and restore configuration that can be accessed from the web and managed centralised.

Keywords: *network automation; python; django; paramiko; network management.*

1. PENDAHULUAN

Perusahaan yang bergerak di bidang jaringan dengan jumlah perangkat jaringan terpasang yang banyak merupakan salah satu tantangan yang harus dihadapi perusahaan dalam melakukan konfigurasi dan perawatan perangkat. Metode tradisional dengan melakukan proses *remote* tiap perangkat akan memakan waktu yang lama. Otomatisasi jaringan merupakan solusi untuk melakukan pekerjaan-pekerjaan yang rumit tersebut dan dapat diimpelementasikan ke perangkat yang mendukung protokol

SSH sehingga pekerjaan bisa diselesaikan jauh lebih cepat dan juga efisien dalam pemeliharaan jaringan dengan prosedur yang lebih mudah diikuti dan diimplementasikan di dalam jaringan berskala besar.

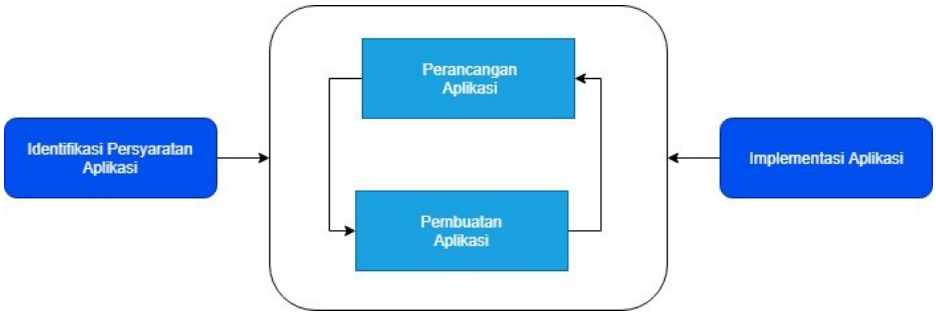
Otomatisasi jaringan menggunakan logika pemrograman untuk memanajemen sumber dan layanan jaringan, hal tersebut dapat memungkinkan teknisi jaringan secara cepat dalam mengkonfigurasi dan mengintegrasikan infrastruktur jaringan (*layer 1 – 3)* dan layanan aplikasi (*layer 4 – 7)* [1]. Salah satu *library* Python yang digunakan sebagai otomatisasi adalah Paramiko. Paramiko merupakan *library* Python yang menggunakan protokol SSHV2 sebagai interaksi dan komunikasi ke perangkat lain yang mendukung SSHv2 [2][3]. Pada penelitian Paul MIHĂILĂ, Titus BĂLAN, Radu CURPEN dan Florin SANDU yang berjudul "*Network Automation and Abstraction using Python Programming Methods*" menunjukkan pentingnya otomatisasi dalam jaringan konvensional yang tidak mendukung protokol *OpenFlow* SDN. Pada jaringan konvensional yang masih menggunakan perangkat jaringan model lama yang diproduksi oleh beberapa *vendor* yang berbeda sangat sulit untuk dikendalikan atau dikontrol, karena perbedaan *syntax* dasar dan belum mendukung protokol *OpenFlow* untuk diterapkan jaringan SDN. Pada penelitian ini dilakukan otomatisasi pembuatan VLAN pada 3 switch Cisco yang dikontrol menggunakan skrip Python. Terdapat dua *library* yang digunakan yaitu Paramiko dan Netmiko. Pada *library* Netmiko penggunaan untuk koneksi ke *switch* menggunakan *ConnectHandler* yang juga menggunakan SSH dibagian *backend*, selain itu Netmiko dapat menentukan tipe perangkat yang akan diatur. Dengan menggunakan Phyton, administrator jaringan tidak perlu mengkonfigurasi sendiri setiap perangkat jaringan. Administrator hanya perlu membuat infrastruktur yang tetap dan dengan menerapkan *scripting* otomatisasi. Otomatisasi dapat menggunakan Phyton dan koneksi *Secure Shell* [4].

Ahmad Rosid Komarudin pada buku berjudul "Otomatisasi Administrasi Jaringan Dengan *Script* Python" menerapkan otomatisasi menggunakan *library* Paramiko, Netmiko, Pyntc, Napalm, dan Ansible dan diterapkan pada perangkat Cisco. Semua *tools* yang digunakan tersebut dapat diimplementasikan ke perangkat Cisco untuk otomatisasi jaringan. Dari percobaan yang telah dilakukan *tools* Napalm dan Ansible merupakan *tools* dengan fitur yang paling *powerfull* untuk diterapkan di perangkat Cisco. Ansible memiliki modul tersendiri untuk jaringan dan mendukung banyak *vendor* perangkat jaringan. Sedangkan dukungan *vendor* Napalm masih terbatas seperti Arista EOS, Cisco IOS, Cisco IOS-XR, Cisco NX-OS dan Juniper JunOS [5].

Pengimplementasian otomatisasi jaringan pada Router OS Mikrotik menggunakan Ansible dilakukan I Made Bayu Swastika dan I Gede Oka Gartria Atitama pada jurnal yang berjudul "Otomatisasi Konfigurasi Mikrotik Router Menggunakan *Software* Ansible". Otomatisasi konfigurasi yang dilakukan yaitu mengatur *bandwidth* menggunakan fitur *Queue Tree* pada Mikrotik. Ditambahkan modul RouterOS API pada Ansible agar *software* Ansible dapat terhubung dan mengkonfigurasi *router* Mikrotik melalui API. Pengujian menggunakan metode *Black-Box Testing* dengan hasil *software* Ansible dapat mengkonfigurasi beberapa *queue tree* secara berurutan dalam sekali eksekusi *file* dengan waktu proses kurang lebih 1 menit untuk 19 konfigurasi *queue tree* [6]. Selain *vendor* Mikrotik Ansible dapat digunakan sebagai otomatisasi konfigurasi pada *vendor* Cisco [7]. Pembuatan otomatisasi jaringan sebagian besar masih menggunakan sebuah skrip, untuk mempermudah otomatisasi maka diperlukan pengembangan aplikasi otomatisasi administrasi jaringan berbasis *website* yang memiliki tampilan atau GUI dan dapat diakses secara terpusat.

2. METODOLOGI PENELITIAN

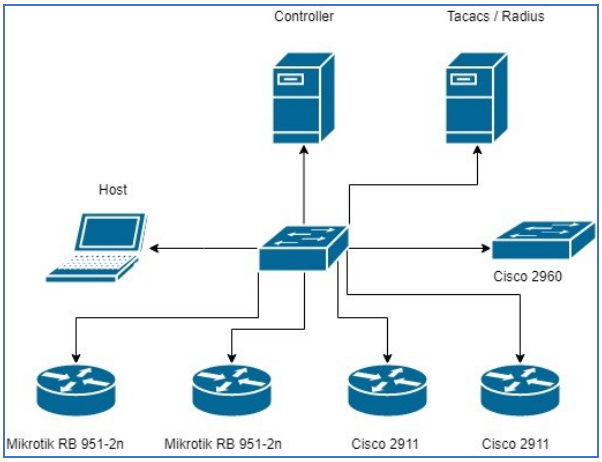
Alur penelitian ini dimulai dari studi literatur dari beberapa jurnal dan buku sebagai sumber referensi penelitian. Pengembangan sistem aplikasi menggunakan metode *Rapid Application Development* (RAD) yang memiliki empat fase utama yaitu fase identifikasi, perancangan, pembuatan dan implementasi aplikasi [8], fase metode RAD dapat dilihat pada Gambar 1.



Gambar 1. Fase Model RAD

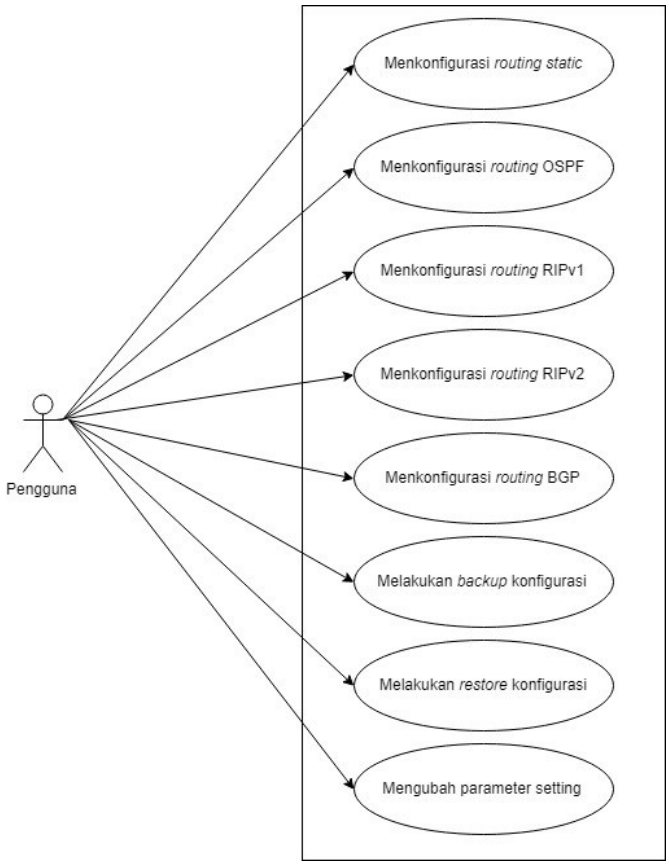
Pada tahap identifikasi persyaratan aplikasi didapatkan kebutuhan perangkat keras dan perangkat lunak yang akan digunakan dalam pengembangan aplikasi, perangkat keras yang digunakan terdiri dari satu unit PC / Laptop, satu unit *server*, dua unit *router* Mikrotik RB951-2n, dua unit *router* Cisco 2911, dua *router* Cisco 2960, satu unit *switch unmanageable.* Perangkat lunak yang digunakan adalah Ubuntu sebagai *server,* Python 3.6.7 versi yang digunakan dalam pengembangan aplikasi, *library* Paramiko 2.4 digunakan sebagai otomatisasi, Django sebagai *web framework,* dan GNS3 2.1 digunakan sebagai *tools* simulasi. Kebutuhan fungsional aplikasi yang didapatkan yaitu sistem aplikasi dapat menambahkan konfigurasi *routing static* dan *dynamic* berupa OSPF, RIPv1, RIPv2, BGP, selanjutnya dapat menambahkan konfigurasi Vlan, melakukan *backup restore* konfigurasi, dan menambahkan konfigurasi *vendor.* Sedangkan kebutuhan non-fungsional sistem aplikasi berupa sistem aplikasi memiliki tampilan antarmuka *user-friendly* dan *responsive,* dapat dijalankan pada beberapa jenis *browser* dan sistem operasi.

Tahapan perancangan aplikasi berupa perancangan topologi, model dan tampilan. Ubuntu sebagai *web server* digunakan sebagai *server* pengembangan sistem aplikasi dimana akan diinstall *web framework* Django, Paramiko dan *library* pendukung lainnya. TACACS+ digunakan sebagai manajemen *user* pada perangkat Cisco [9]. Radius *server* digunakan untuk manajemen *user* pada perangkat Mikrotik. Berikut topologi jaringan pada Gambar 2.



Gambar 2. Topologi Jaringan Aplikasi

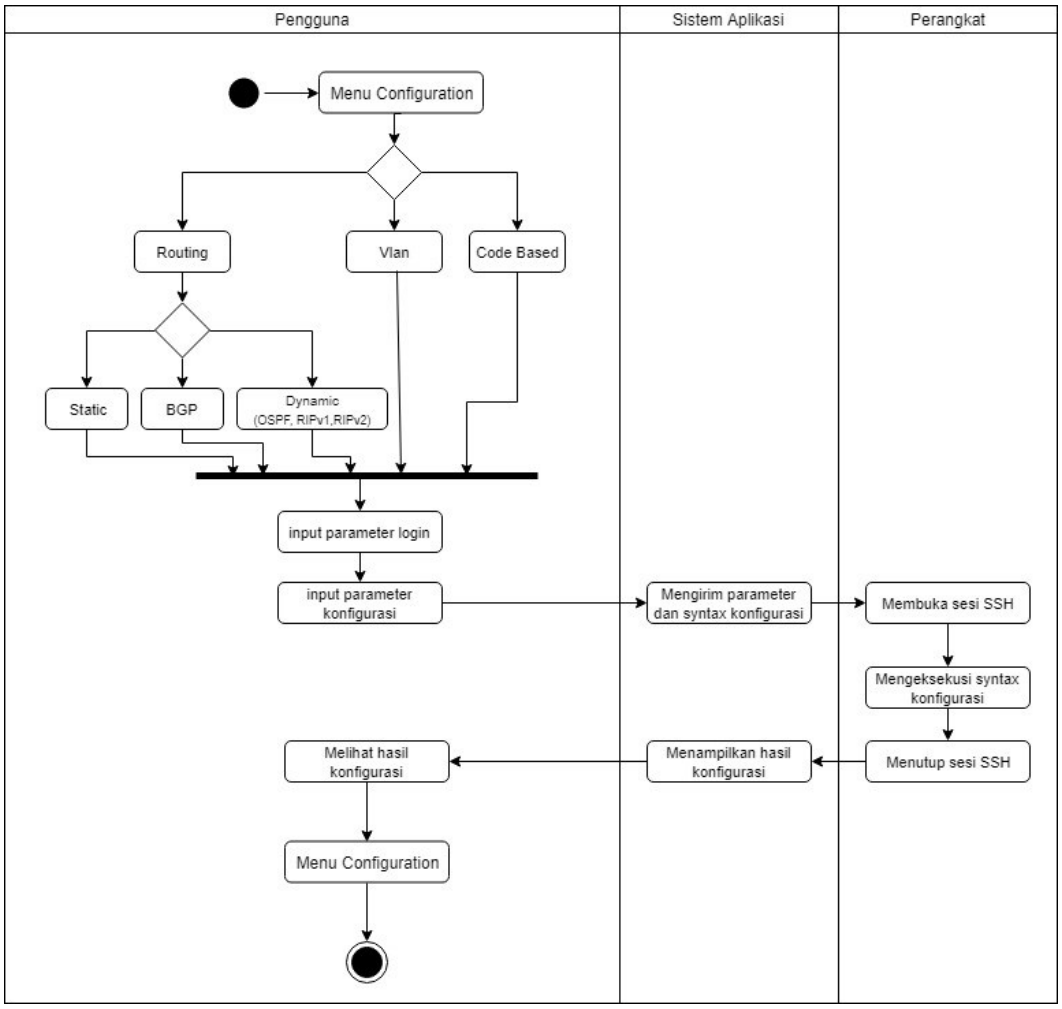
Pada tahap perancangan model dilakukan dengan menggunakan diagram UML. Pada penelitian ini diagram UML yang digunakan adalah *use case diagram*, dan *activity diagram*. *Use case diagram* digunakan untuk merepresentasikan sebuah interaksi antara *user* dengan sistem dan membuat visualisasi dari fungsi yang dibuat dalam aplikasi [10]. Gambar 3 merupakan *use case diagram* sistem aplikasi yang dikembangkan.



Gambar 3. *Use Case Diagram* Aplikasi

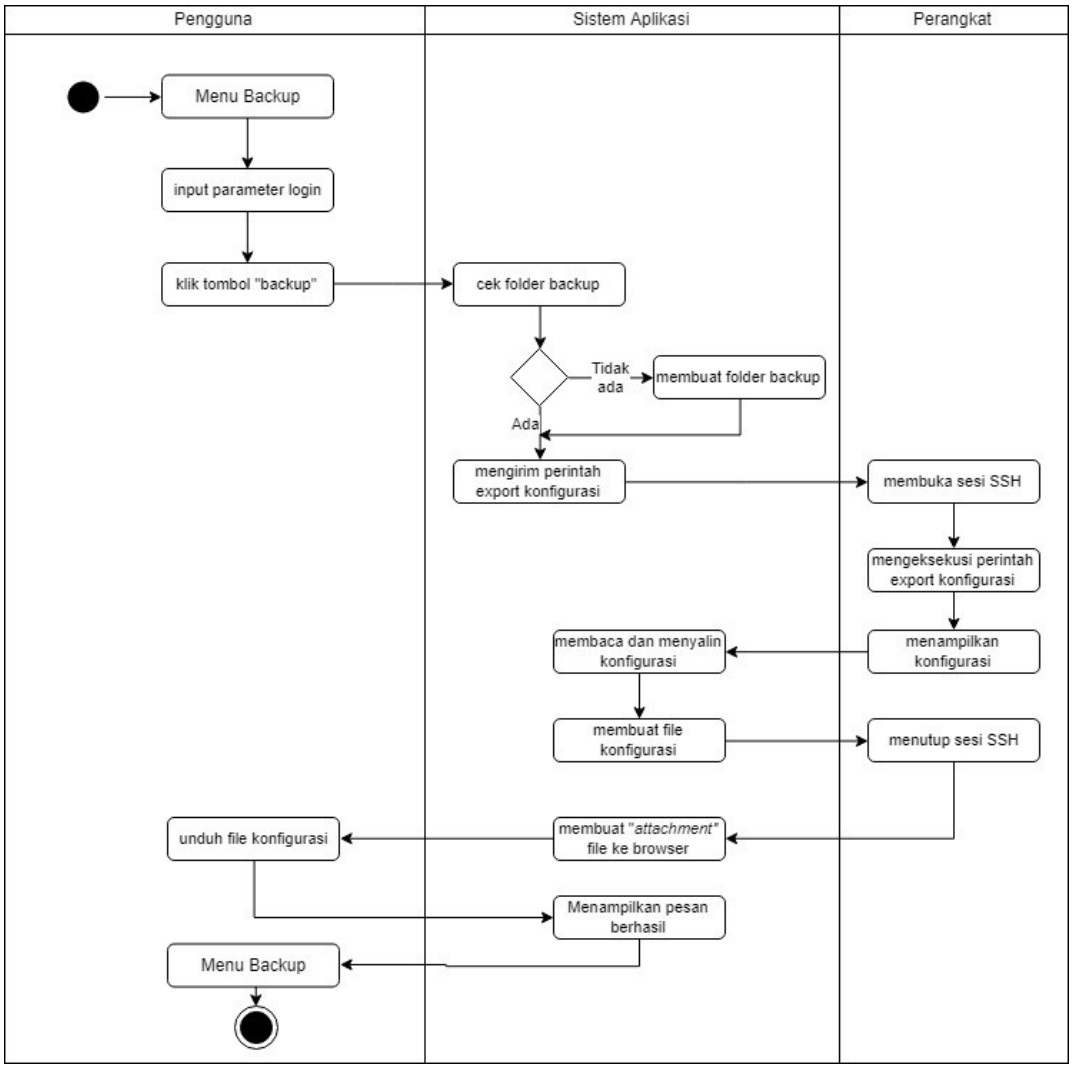
Activity Diagram merupakan diagram UML yang digunakan untuk menggambarkan aliran kerja dan aktivitas yang dapat dilakukan oleh lingkup sistem [11]. Dalam perancangan *activity diagram*, seluruh aktivitas yang terdapat di dalam sistem aplikasi ditampilkan sesuai dengan lima fitur utama yang terdapat pada aplikasi.

Pada *activity diagram* fitur *Configuration* terdapat tiga objek yang memiliki aktivitas masing-masing. Objek pengguna mempunyai aktivitas dalam memilih konfigurasi dan mengisi parameter konfigurasi, objek sistem aplikasi memiliki tanggung jawab dalam aktivitas mengirim konfigurasi dan perangkat sebagai objek yang dikonfigurasi. Pada Gambar 4 terlihat *activity diagram* fitur *Configuration*.



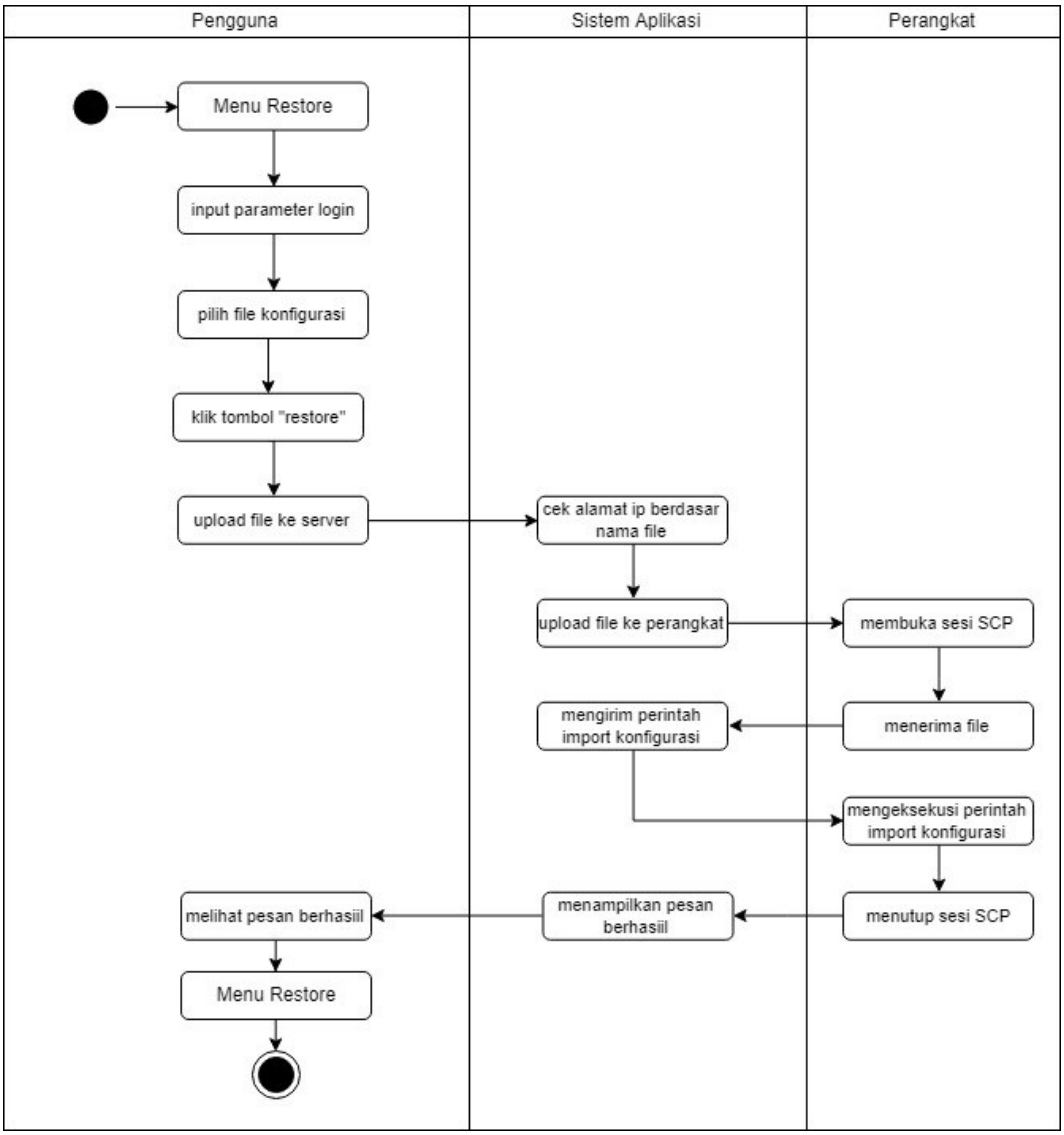
Gambar 4. ***Activity Diagram*** **Fitur** ***Configuration***

Activity diagram fitur *Backup* terdapat tiga objek yang memiliki aktivitas masing-masing. Objek pengguna mempunyai aktivitas dalam mengisi parameter login ke perangkat yang dituju, objek sistem aplikasi memiliki tanggung jawab dalam aktivitas membuat *directory backup* yang berfungsi sebagai *directory* tempat penyimpanan *file backup* sementara, selain itu melakukan aktivitas dalam mengirim *syntax* konfigurasi. Perangkat sebagai objek yang akan dilakukan *backup* konfigurasi. Berikut *activity diagram backup configuration* yang terdapat pada Gambar 5.



Gambar 5. ***Activity Diagram*** **Fitur** ***Backup***

Pada *Activity diagram* fitur *Restore* memiliki tiga objek yang memiliki aktivitas masing-masing. Objek pengguna mempunyai aktivitas dalam mengisi parameter login ke perangkat yang dituju dan memilih *file* konfigurasi yang akan digunakan sebagai *file restore*, objek sistem aplikasi memiliki tanggung jawab dalam aktivitas mengirim *file* konfigurasi dan mengirim *syntax* konfigurasi. Perangkat sebagai objek yang akan dilakukan *restore* konfigurasi. Activity diagram *restore configuration* terdapat di Gambar 6.



Gambar 6. *Activity Diagram* Fitur *Restore*

Tahap selanjutnya yaitu pembuatan aplikasi. Pengembangan aplikasi otomatisasi administrasi jaringan menggunakan *library* utama Paramiko yang digunakan sebagai otomatisasi. *Library* Paramiko akan menghubungkan *server* dengan perangkat jaringan melalui protokol SSH. Dalam pengembangan aplikasi berbasis *website* menggunakan *web framework* Django dimana pada bagian *backend* sistem aplikasi menggunakan bahasa Python. Sedangkan pada bagian *frontend* atau tampilkan menggunakan HTML, *Cascading Style Sheet* (CSS) dan JavaScript

Tahapan selanjutnya adalah tahap impelementasi aplikasi. Sebelum diimplementasikan secara *real*, dilakukan terlebih dahulu secara simulasi menggunakan GNS3. Implementasi pada jaringan *real* dilakukan dengan menggunakan *router* MikroTik RB951, Cisco 2911 dan *switch* Cisco 2960. Pengujian dilakukan sesuai dengan skenario yang telah dibuat. Pengujian menggunakan metode *Black-Box Testing* untuk mengetahui keberhasilan fungsi sistem aplikasi dapat berjalan dengan baik atau tidak [12].

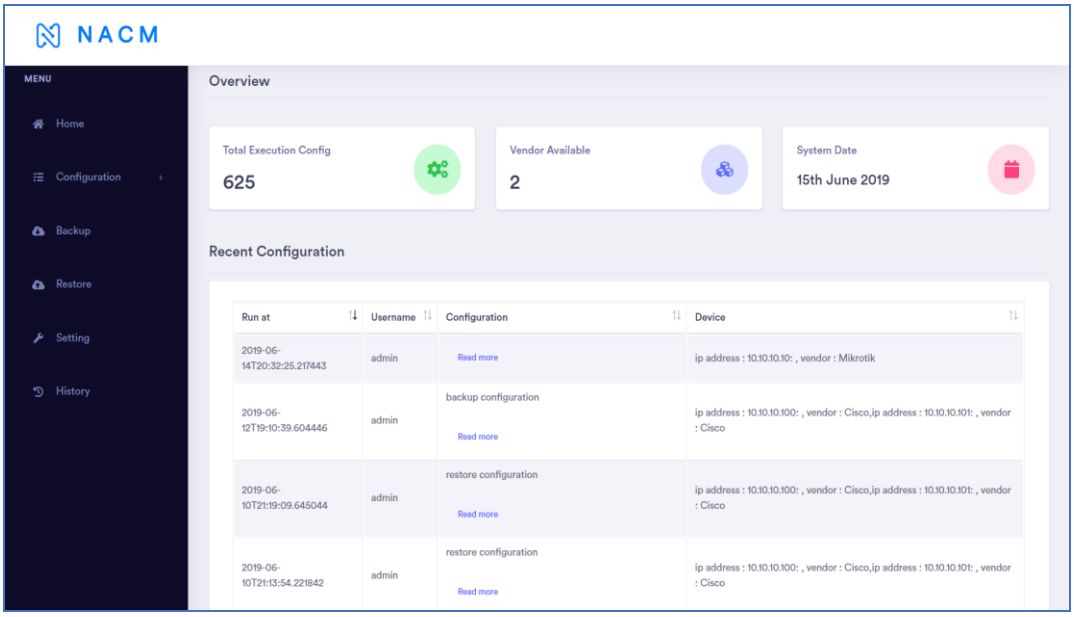
3. HASIL DAN PEMBAHASAN

3.1 Tampilan Aplikasi

Sistem aplikasi yang dikembangkan memiliki nama *Network Automation Configuration Management* atau disingkat NACM. Tampilan antarmuka sistem aplikasi dikembangkan berdasarkan dari rancangan tampilan yang telah dibuat. Tampilan antarmuka sistem aplikasi terdiri dari tampilan daftar menu, tampilan halaman utama sistem aplikasi, tampilan *submenu Routing static, Routing dynamic, Routing BGP,* Vlan, *Codebased, Backup, Restore,* dan *History.* Pengembangan antarmuka aplikasi

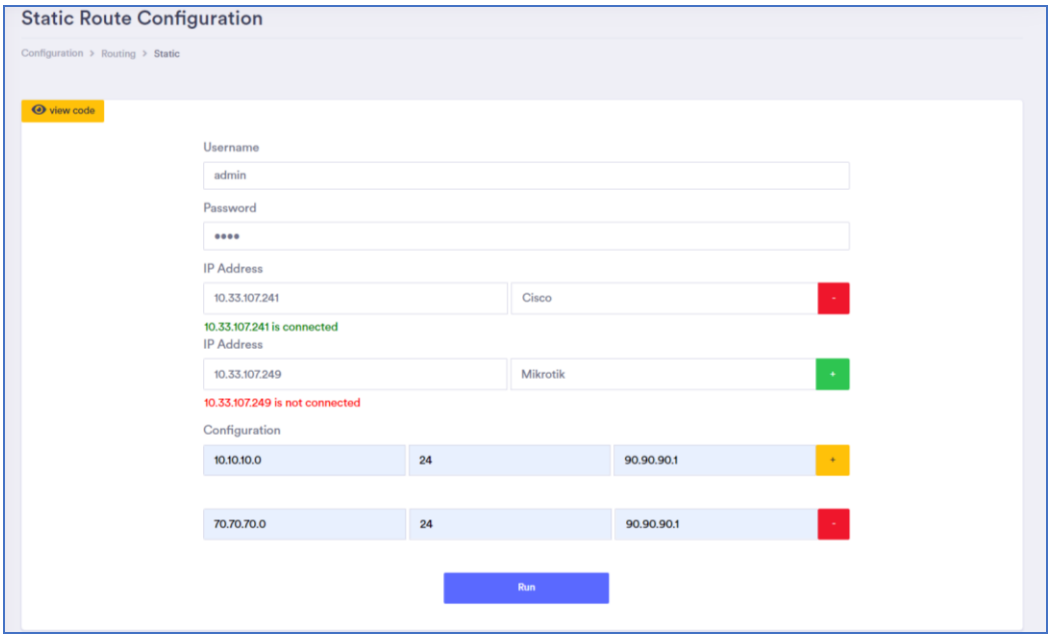
menggunakan CSS *framework* Bootstrap untuk mempermudah dalam pembuatan antarmuka sistem aplikasi. Hasil yang didapatkan sistem aplikasi memiliki antarmuka yang responsif dimana ukuran tampilan aplikasi akan mengikuti ukuran layar pengguna

Pada halaman utama atau beranda sistem aplikasi terdapat *Overview* yang menampilkan informasi jumlah konfigurasi yang telah dieksekusi, jumlah *vendor* yang telah dilakukan *setting* dimana data-data tersebut diambil dari *database* SQLite. Bagian *Recent Configuration* menampilkan informasi 5 konfigurasi terakhir yang dieksekusi. Terdapat informasi waktu konfigurasi dieksekusi yang terdiri dari tahun, bulan, tanggal, dan jam eksekusi. Informasi *username* yang melakukan konfigurasi, selanjutnya informasi konfigurasi yang dieksekusi baik konfigurasi *routing,* Vlan, *code based, backup,* dan *restore.* Selain itu terdapat informasi *device* yang dilakukan konfigurasi terdiri dari alamat ip *device* dan tipe *vendor device.* Informasi yang ditampilkan pada halaman utama sistem aplikasi bertujuan sebagai *log* serderhana sehingga dapat ditelusuri pengguna yang melakukan konfigurasi ke perangkat jaringan menggunakan sistem aplikasi. Tampilan halaman utama sistem aplikasi dapat dilihat pada Gambar 7.



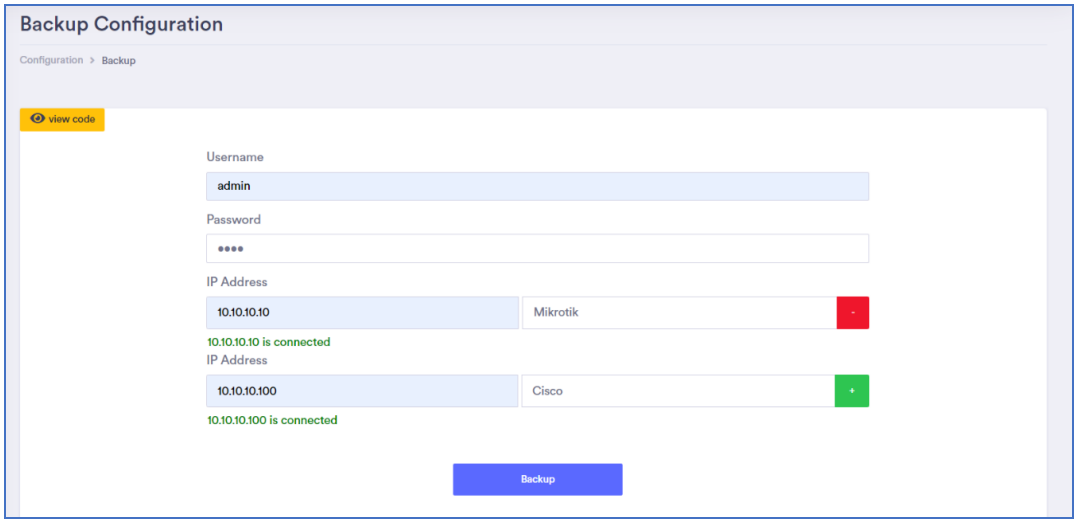
Gambar 7. Tampilan Halaman Utama Aplikasi

Pada fitur *configuration* terdapat beberapa *submenu* seperti *routing static, dynamic,* BGP, Vlan, dan *Code Based.* Pada setiap *submenu* terdapat kolom pengisian identitas *username* dan *password* SSH, selain itu terdapat kolom pengisian alamat ip dan *vendor* perangkat tujuan. Setiap *submenu* memiliki kolom parameter konfigurasi masing-masing sesuai dengan konfgurasi yang akan digunakan, seperti pada fitur *routing static* terdapat kolom parameter *static routing* yaitu *destination, prefix,* dan *gateway*. Berikut salah satu tampilan fitur *configuration* yaitu *routing static* pada Gambar 8.



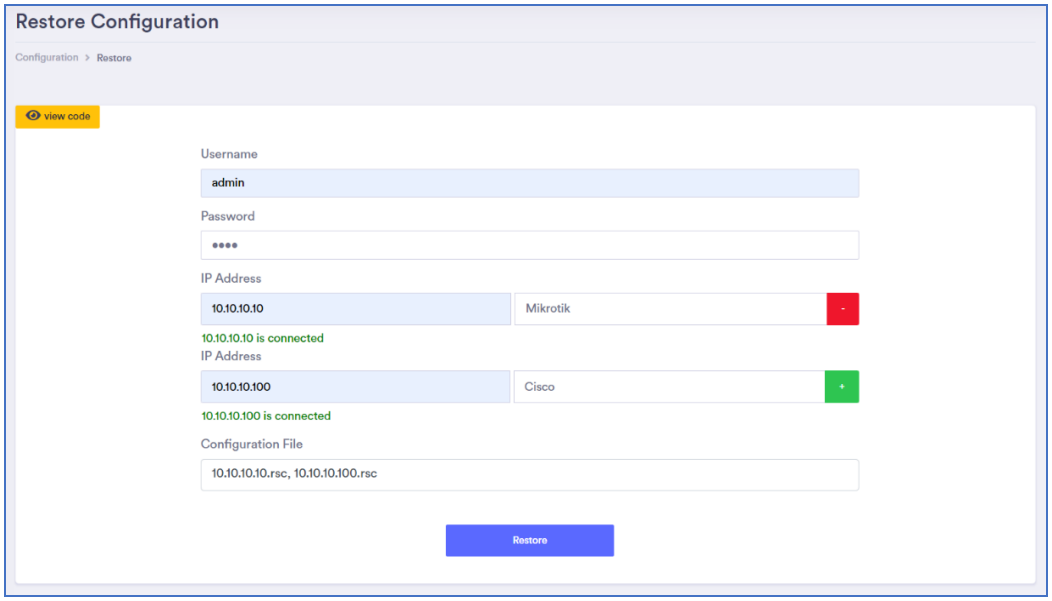
Gambar 8. Tampilan Halaman Fitur *Configuration Routing Static*

Parameter kolom *input* pada fitur *backup* hanya berupa kolom *username, password, ip address,* dan *vendor*. Setelah pengguna menekan tombol *"Backup"* akan muncul jendela *pop-up* yang menampilkan informasi proses pengunduhan *file* konfigurasi. Pada jendela *pop-up* terdapat tombol *"Finish"* untuk melakukan *reload* halaman ketika proses unduh selesai. Berikut tampilan fitur *backup* pada Gambar 9.



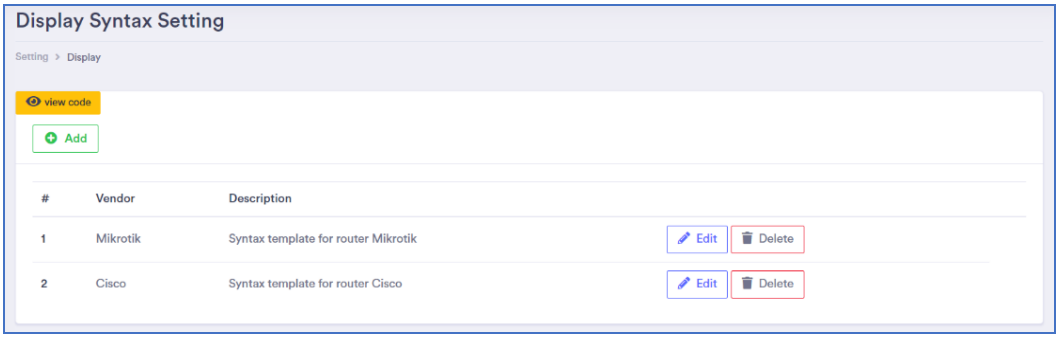
Gambar 9. Tampilan Halaman Fitur *Backup*

Fitur *restore* memiliki kolom *Configuration File* untuk memilih *file* konfigurasi yang akan digunakan sebagai *file restore* ke perangkat tujuan. Ketika pengguna menekan kolom tersebut akan muncul jendela *file* untuk memilih *file* konfigurasi. Adapun tampilan pada fitur *restore* terdapat pada Gambar 10.



Gambar 10. Tampilan Halaman Fitur *Restore*

Pada fitur *Setting* menampilkan informasi *vendor* yang telah diatur parameter *syntax* konfigurasi perangkat. Terdapat informasi nama *vendor, description* dan tombol add untum menambah *setting vendor,* tombol edit untuk mengubah parameter *syntax vendor* dan tombol *delete* untuk menghapus *setting vendor.* Berikut antarmuka fitur *setting* pada Gambar 11.



Gambar 11. Tampilan Halaman Fitur *Setting*

3.2 Hasil Black-Box Testing

Pengujian menggunakan metode *Black-Box Testing* pada penelitian merupakan rangkuman hasil dari semua pengujian. Pengujian dilakukan sesuai dengan Tabel 3.2 *Test Script* Pengujian Aplikasi. Hasil pengujian yaitu semua fitur sistem aplikasi dan *case* dapat dijalankan. Hal ini menunjukkan bahwa fungsi yang dikembangkan pada sistem aplikasi berhasil berfungsi dengan baik. Hasil *Black-Box Testing* dapat dilihat pada Tabel 1.

Tabel 1. Hasil *black-box testing*

No.	*Skenario*	*Kasus Pengujian*	*Hasil*
1	Fitur *Settings*	Menambah dan mengubah parameter Settings	Berhasil
2	Fitur *Configuration*	Menambahkan konfigurasi *static routing*	Berhasil
3		Menambahkan konfigurasi *dynamic routing* OSPF	Berhasil
4		Menambahkan konfigurasi *dynamic routing* RIP versi 1	Berhasil
5		Menambahkan konfigurasi *dynamic routing* RIP versi 2	Berhasil
6		Menambahkan konfigurasi *dynamic routing* BGP	Berhasil
7		Menambahkan konfigurasi Vlan	Berhasil
8		Membuat *banner message of the day* (MOTD)	Berhasil
9	Fitur *Backup*	Melakukan *backup* konfigurasi perangkat	Berhasil
10	Fitur *Restore*	Melakukan *restore* konfigurasi perangkat	Berhasil

Pada skenario fitur *Settings* dengan pengujian menambah dan mengubah parameter *setting routing static*, *dynamic*, VLAN, *backup*, dan *restore* dengan *vendor* Cisco dan Mikrotik hasilnya parameter konfigurasi berhasil disimpan dan dapat dieksekusi melalui sistem aplikasi. Skenario fitur *Configuration* dengan kasus menambahkan konfigurasi *static routing*, *dynamic routing* berupa OSPF, RIPv1, RIPv2, BGP, penambahan konfigurasi VLAN dan pengujian *menu code based* menggunakan uji konfigurasi *banner message of the day* (MOTD) hasil pengujiannya dinyatakan berhasil, dilakukan pengecekan bahwa konfigurasi telah masuk pada perangkat yang dituju dan dapat dijalankan. Pada skenario fitur *backup* hasil pengujiannya berhasil, konfigurasi perangkat yang dituju berhasil di*backup* ke PC yang mengakses sistem aplikasi dan isi *backup* konfigurasi sesuai dengan konfigurasi perangkat yang di*backup*. Pada skenario fitur *restore, file restore* berhasil masuk dan diaplikasikan ke perangkat yang dituju sehingga dinyatakan berhasil.

4. KESIMPULAN

Berdasarkan data yang diperoleh serta analisis yang sudah dilakukan dari penelitian Pengembangan Aplikasi Otomatisasi Administrasi Jaringan Berbasis Website Menggunakan Bahasa Pemrograman Python dapat ditarik kesimpulan sebagai berikut:

a. Telah dikembangkan aplikasi otomatisasi administrasi jaringan berbasis *website* menggunakan *library* utama Paramiko dan *web framework* Django.
b. Pengembangan aplikasi menghasilkan lima buah fitur yaitu konfigurasi *Routing*, Vlan, Backup, *Restore,* dan Setting. Dimana pada fitur-fitur tersebut dapat dilakukan fungsi utama aplikasi dalam melakukan konfigurasi administrasi jaringan berupa *routing static, dynamic* OSPF, RIPv1, RIPv2, BGP, *backup* dan *restore* konfigurasi.
c. Metode *Black-box* Testing sebagai pengujian aplikasi menunjukkan bahwa semua fungsi pada aplikasi yang dikembangkan pada penelitian ini berfungsi dengan baik dan berhasil diterapkan pada vendor yang berbeda yaitu Cisco dan Mikrotik.

DAFTAR PUSTAKA

[1] Anonim. 2018. *Network Automation for Everyone.* Redhat.
[2] Chou, E. 2017. Mastering Python networking. 2nd ed. packt. Birmingham.
[3] Forcier, J. 2018. Paramiko's documentation. [Online]. http://docs.paramiko.org/en/2.4/. Diakses pada tanggal 1 Desember 2018.
[4] Mihăilă, P., Bălan, T., Curpen, R. dan Sandu, F. 2017. Network Automation and Abstraction using Python Programming Methods. Electronics and Computer Department. Transilvania University.
[5] Komarudin, A. 2018. Otomatisasi Administrasi Jaringan Dengan Script Python. Jasakom. Jakarta.
[6] Swastika, I., Atitama, G. 2017. Otomatisasi Konfigurasi Mikrotik Router Menggunakan Software Ansible. Fakultas Matematika dan Ilmu Pengetahuan Alam. Universitas Udayana.
[7] Wijaya, J. 2018. Network Automation using Ansible for Cisco Router Basic Configuration. Teknik Elektro dan Informatika. Institut Teknologi Bandung.
[8] Martin, James. 1991. Rapid Application Development. Macmillan Publishing Co., Inc,.

[9] Hermawan, E. 2017. Aplikasi Radius atau Tacacs+ Server Untuk Mengatur dan Mengontrol Penggunaan Jaringan Komputer di Politeknik Negeri Balikpapan. Jurusan Teknik Elektronika. Politeknik Negeri Balikpapan.
[10] Awaad, M.H, H, Krauss, H. D. Schmatz. 2005. Advanced Praise for The Unified Modeling Language Reference Manual, Second Edition, vol. 240, no. 3.
[11] A.S, Rossa dan M. Shalahuddin, "Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek", Penerbit Informatika, Bandung, 2013
[12] Patton, Ron. 2003. Software Testing. SAMS Publishing.